**Aula VI - DevAppII**

**Lógica em Dart**

**Estruturas de Repetição em Dart**

**For (para)**

**For in**

**While (enquanto)**

**Do While (faça enquanto)**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Fontes:**

<https://dart.dev/>

<https://www.devmedia.com.br/conceitos-e-exemplos-do-for-lacos-de-repeticoes-estrutura-da-linguagem-parte-2/18871>

<https://www.arquivodecodigos.com.br/dicas/3760-dart-usando-o-laco-loop-for-da-linguagem-dart.html>

<https://gist.github.com/nunomazer/0a83700c16996e18cb8e05cb653065b6>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Estruturas de Repetição**

**O que são estruturas de repetição na programação?**

As estruturas de repetição ou laços de repetição, também conhecidos como laços de iteração ou simplesmente loops, são comandos que permitem iteração de código, ou seja, que comandos presentes no bloco sejam repetidos diversas vezes. Através de laços de repetição é possível criar programas que percorrem dados, listas e arrays, analisando individualmente cada elemento, e até mesmo criar trechos de código que sejam repetidos até que certa condição estabelecida seja cumprida.

**São elas:**

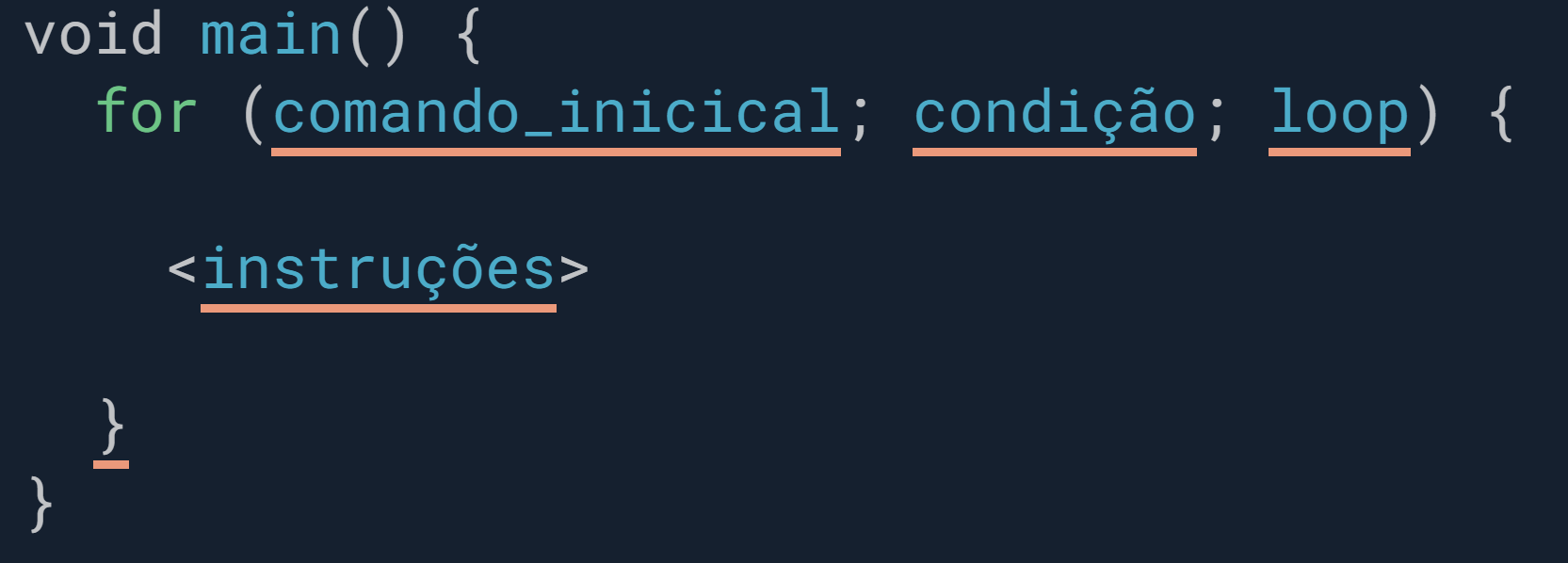
* **For (para)**
* **For in (Arrays e Listas)**
* **While (enquanto)**
* **Do While (faça enquanto)**

|  |
| --- |
| **Resumindo!**  **As estruturas de repetição também são conhecidas como laços (loops) e são utilizadas para executar, repetidamente, uma instrução ou bloco de instrução enquanto determinada condição estiver sendo satisfeita.**  **Qualquer que seja a estrutura de repetição, ela contém quatro elementos fundamentais: inicialização, condição, corpo e iteração.**  **A inicialização compõe-se de todo código que determina a condição inicial da repetição.**  **A condição é uma expressão booleana avaliada após cada leitura do corpo e determina se uma nova leitura deve ser feita ou se a estrutura de repetição deve ser encerrada.**  **O corpo compõe-se de todas as instruções que são executadas repetidamente.**  **A iteração é a instrução que deve ser executada depois do corpo e antes de uma nova repetição.** |

**For:**

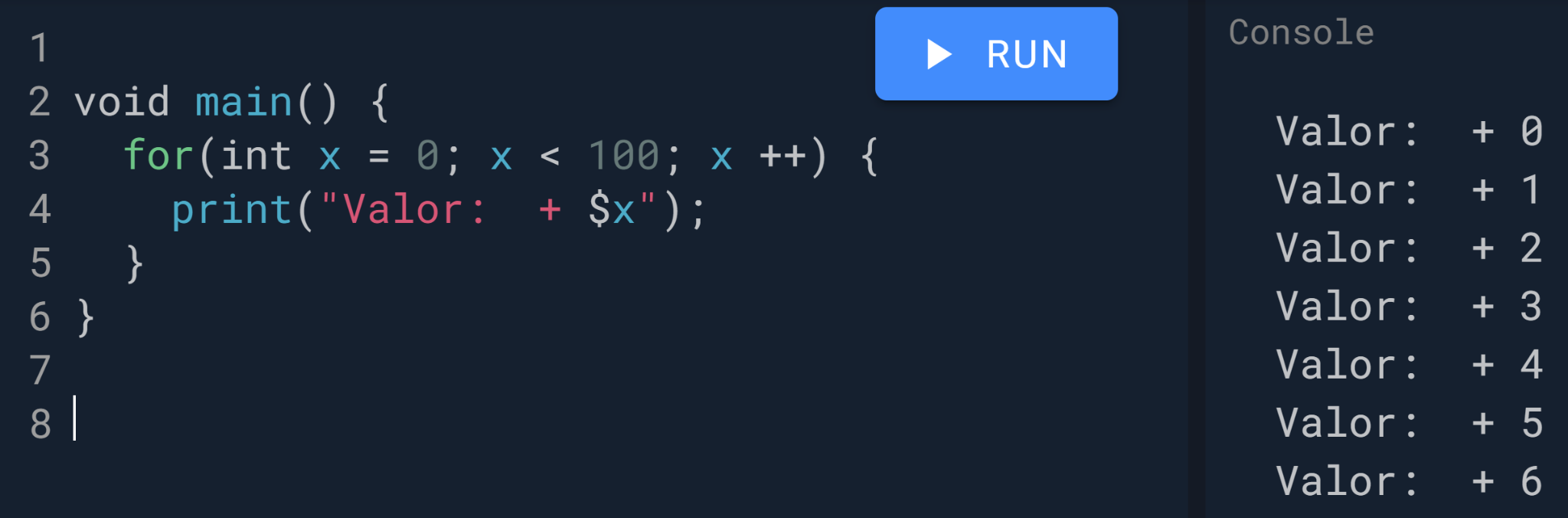
O for (**para, em inglês - quer dizer: “para este caso... fazer...”**) pode conter apenas uma instrução no seu corpo. Neste caso não é necessário abrir um bloco. Isso é assim porque o “for” já implementa alguns comandos na sua assinatura, ou seja, no seu cabeçalho, como a inicialização da variável e o passo da repetição, ou seja, o incremento/decremento da variável (executado sempre no fim de cada ciclo). **O laço for é uma estrutura de repetição compacta.** Seus elementos de **inicialização, condição e iteração são reunidos na forma de um cabeçalho e o corpo é disposto em seguida.**

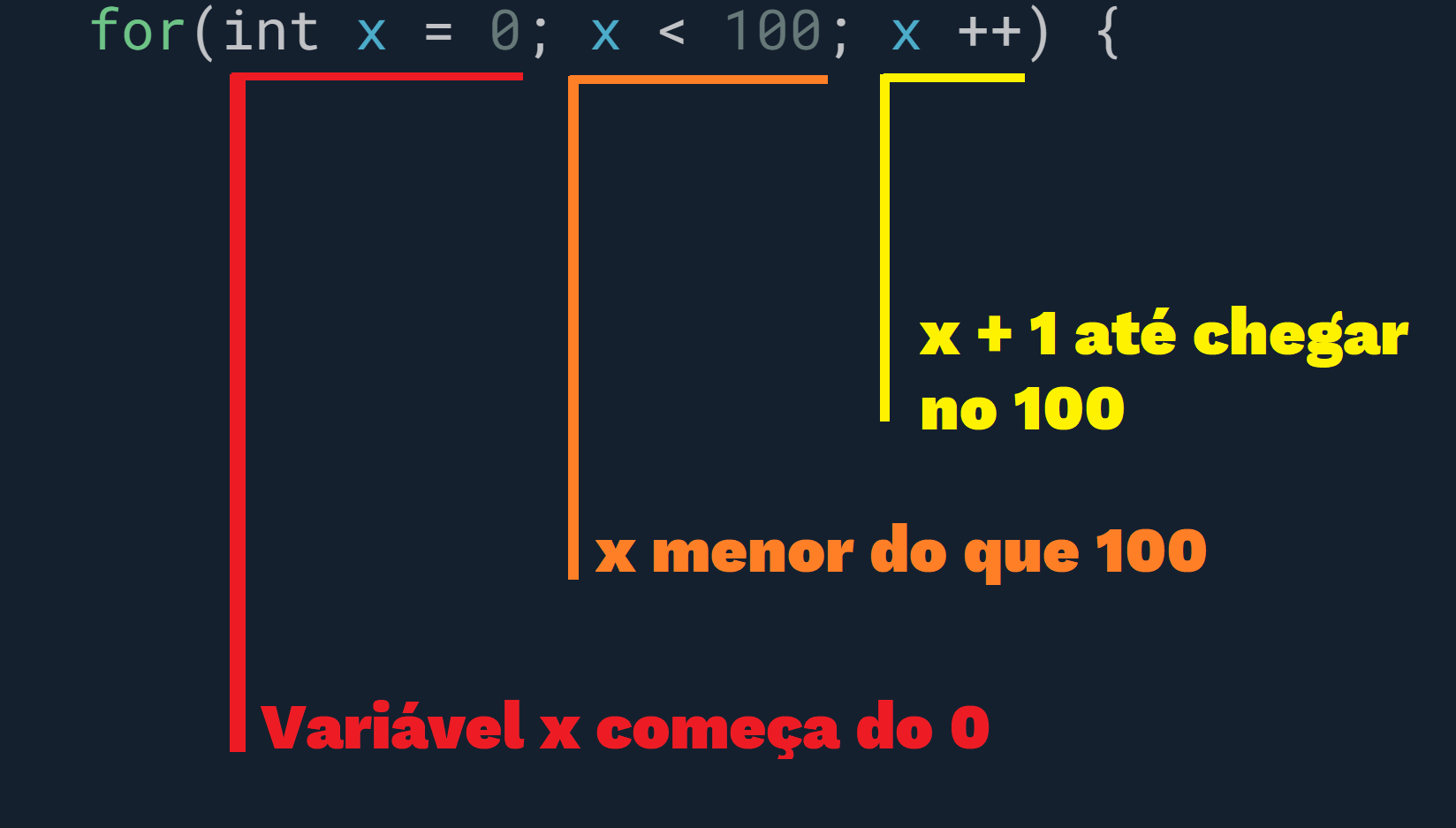
O **laço for e o laço while são apenas formas diferentes de uma mesma estrutura básica de repetição.** Qualquer laço for pode ser transcrito em termos de um laço while e vice-versa. Do mesmo modo que em um laço while, se a condição de um laço for já é falsa logo na primeira avaliação que se fizer dela, as instruções contidas em seu corpo jamais serão executadas.



**Comando inicial**: Este comando executa apenas uma vez, quando o código entra no loop. Aqui podemos declarar uma variável, por exemplo, como é muito comum.

**Condição:** O mesmo tipo de condição que montamos em while e do while. Se ela for verdadeira, as instruções serão executadas. Se for falsa, o loop encerra.

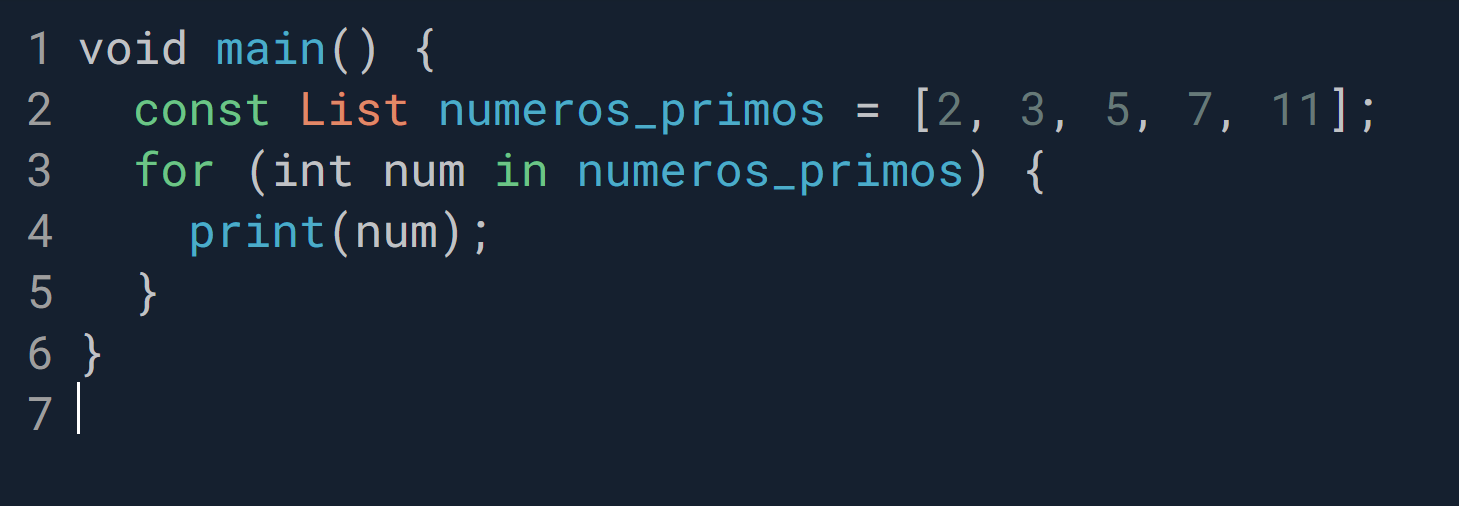
**Loop:** Este comando é executado sempre que o for completa uma volta, ou seja, na primeira execução ele pula este comando.

Sendo que:

**Operadores de Incremento e Decremento**

**For in:**

A estrutura de loop for in tem um nome muito parecido com o For, **porém são estruturas distintas. Enquanto o loop For utiliza um contador para gerenciar as voltas do loop, o for−in faz uso de um contador intrínseco para gerenciar as voltas do loop baseada nas posições de uma coleção (Listas, sets e mapas). Isso faz dele uma estrutura de loop destinada para interações em coleções.**

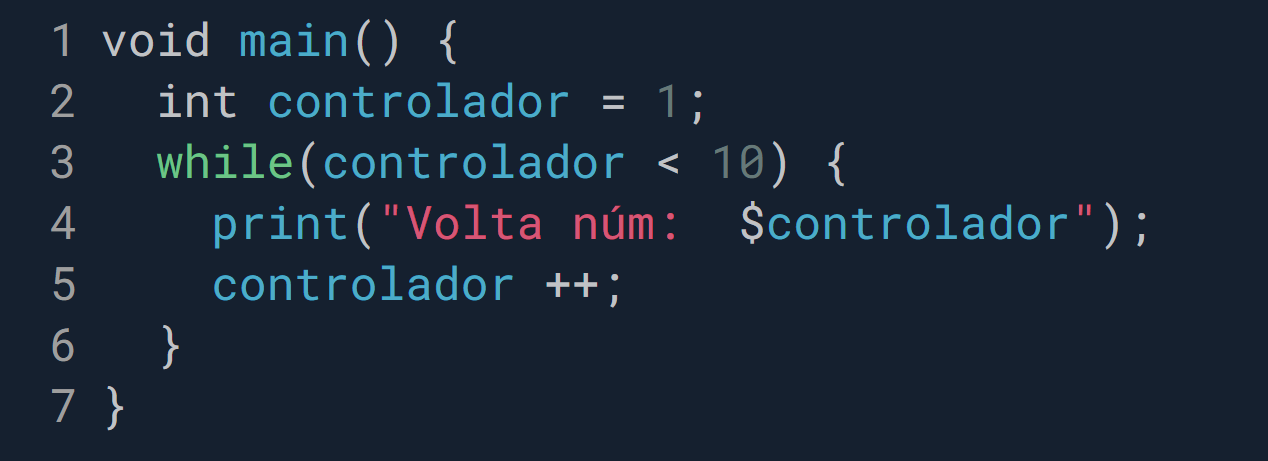
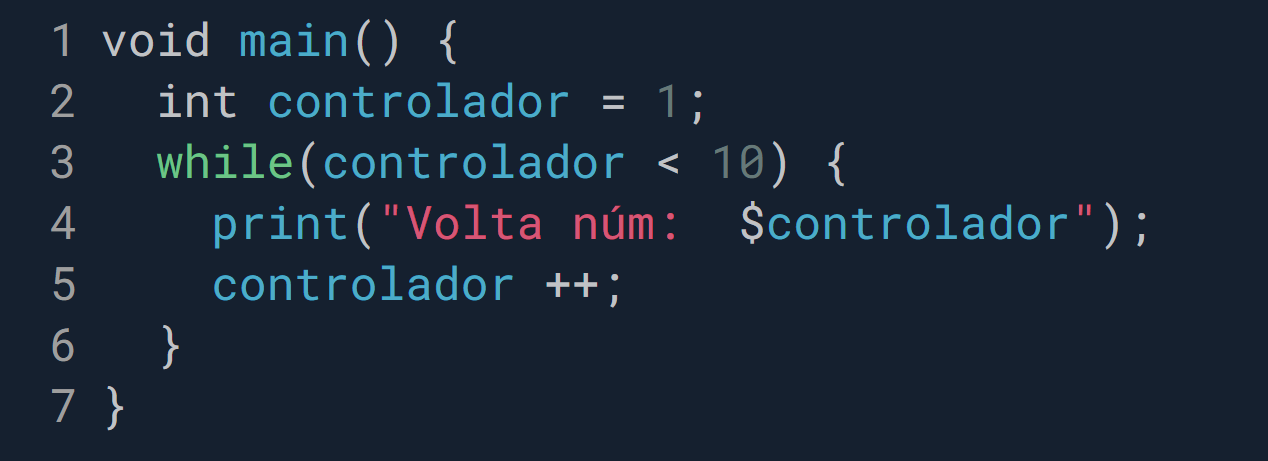


**While (enquanto):**

**O laço de repetição while caracteriza-se por ter seu teste de execução antes de iniciar o loop.** Neste tipo de laço nem sempre temos a execução dos comandos, ou seja, nem sempre ele entra no loop, traduzindo, o **While é utilizado para construir uma estrutura de repetição que executa, repetidamente, uma única instrução ou um bloco delas “enquanto” uma expressão booleana for verdadeira.**

|  |
| --- |
| **Sintaxe básica:**  **enquanto a condição for verdadeira {**  **executa as instruções**  **}** |

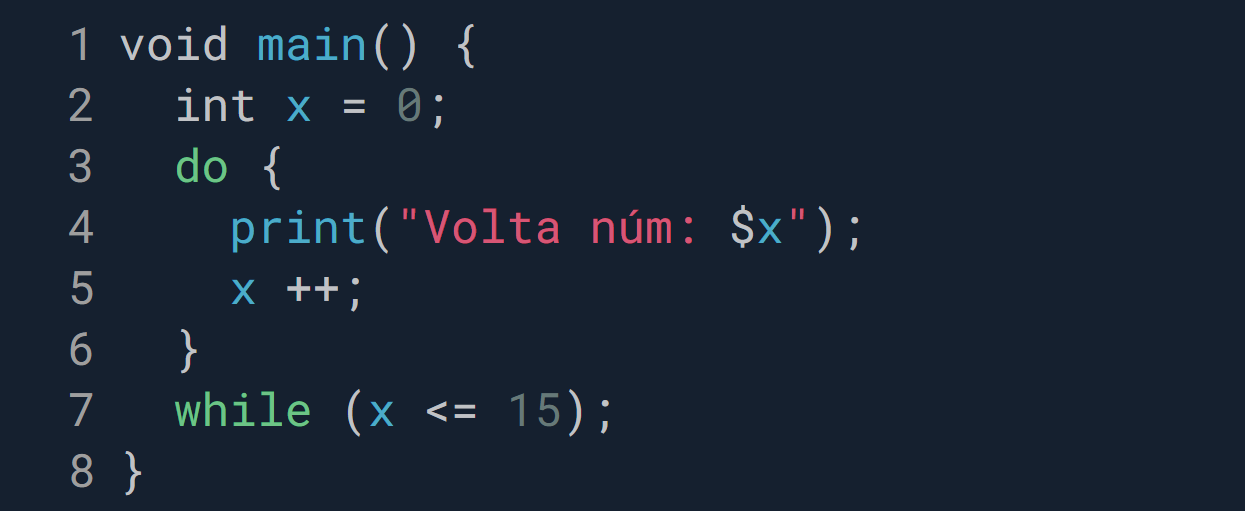
**Exemplos:**



**Do While (faça enquanto):**

A estrutura de repetição **do-while** **é uma variação da estrutura while**. Existe uma diferença sutil, porém importante, entre elas. **Em um laço while, a condição é testada antes da primeira execução das instruções que compõem seu corpo. Desse modo, se a condição for falsa na primeira vez em que for avaliada, as instruções desse laço não serão executadas nenhuma vez. Em um laço do-while, por outro lado, a condição somente é avaliada depois que suas instruções são executadas pela primeira vez, assim, mesmo que a condição desse laço seja falsa antes de ele iniciar, suas instruções serão executadas pelo menos uma vez.**

**Condição:** **O do while executa as instruções enquanto a condição nos parênteses for verdadeira!** No momento que a condição for falsa ele para de executar as instruções dentro do bloco do while.

****